

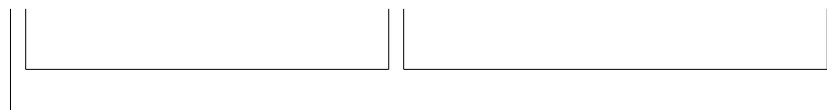
# **Web services**

## Semester challenge Presentation

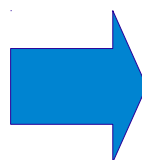
Alexander Betz  
9.7.2013

# Disulfinder

FVREI**C**IQVLMF**C**TLYKLST**C**HRDG



-----	
DISULFIND	
Cysteines Disulfide Bonding State and Connectivity Predictor	
-----	
Query Name: Q30201	
AA	.....10.....20.....30.....40.....50.....60.....70.....
DB_state	MGPRARPALLLLMLLQTAVLQGRLLRSHSLHYLFMGASEQDLGLSLFEALGYVDDQLFVFDHESRRVEPRTPWVSSRI
DB_conf	
DB_flip	
AA	80.....90.....100.....110.....120.....130.....140.....150.....
DB_state	SSQMWLQLSLSLKGMDHMFVDFWTIMENHNHKSESHTLQVILGCEMQEDNSTEGYWKYGYDGDHLEFCPTLDWRAA
DB_conf	0 0
DB_flip	* 4
AA	.160.....170.....180.....190.....200.....210.....220.....230.....
DB_state	EPRAWPTKLEMERHKIRARQNRAYLERDCPAQLQLLELGRGVLDQQVPLVKVTHHVTSSVTLRCRALNYYPNQITM
DB_conf	0 0
DB_flip	1 1



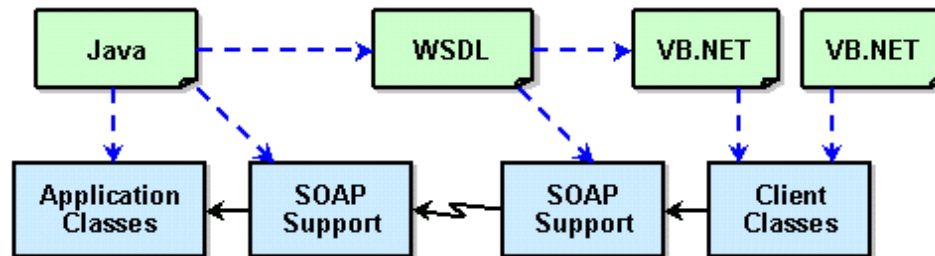
```

1 <?xml version="1.0" encoding="UTF-8" ?>
2 <disulfideBridgePrediction xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:
3   disulfinder="http://i12r-tbl.informatik.tu-muenchen.de/~alex/disulfinder_output.xsc"
4   <bx:referenceSequence>
5     <bx:sequenceRecord xsi:type="bx:GeneralAminoacidSequenceRecord">
6       <bx:sequence>MGPRARPALLLLMLLQTAVLQGRLLRSHSLHYLFMGASEQDLGLSLFEALGYVDDQLFVFDHESRRVEPRTPWVSSRI
7     </bx:sequenceRecord>
8   </bx:referenceSequence>
9   <bx:blockWithOccurrenceReferences>
10    <bx:method name="disulfinder" localId="M#di">
11      <bx:categoryConcept conceptUri="http://edamontology.org/operation_1850" ontologyName="disulfinder"
12      <bx:citation date="2013-06-28"/>
13    </bx:method>
14    <bx:scoreType localId="S#st"/>
15    <bx:scoreType localId="S#co"/>
16    <bx:scoreType localId="S#fl"/>
17    <bx:annotation>
18      <bx:feature>
19        <bx:name>Cystein disulfide bonding</bx:name>
20        <bx:equalConcept term="Binary cystein bonding map inside of one chain"/>
21      </bx:feature>
22      <bx:condensedReferences>
23        <bx:methodIdRef>M#di</bx:methodIdRef>
24      </bx:condensedReferences>
25      <bx:occurrence>
26        <bx:position xsi:type="bx:SequencePosition">
27          <bx:point xsi:type="bx:SequencePoint" pos="123"/>
28        </bx:position>
29        <bx:score value="0" scoreTypeIdRef="S#st"/>
30        <bx:score value="0" scoreTypeIdRef="S#co"/>
31        <bx:score value="1" scoreTypeIdRef="S#fl"/>
32      </bx:occurrence>
33      <bx:occurrence>
34        <bx:position xsi:type="bx:SequencePosition">
35          <bx:point xsi:type="bx:SequencePoint" pos="148"/>
36        </bx:position>
37        <bx:score value="0" scoreTypeIdRef="S#st"/>
38        <bx:score value="4" scoreTypeIdRef="S#co"/>
39        <bx:score value="0" scoreTypeIdRef="S#fl"/>
40      </bx:occurrence>
41    </bx:blockWithOccurrenceReferences>
42  </disulfideBridgePrediction>

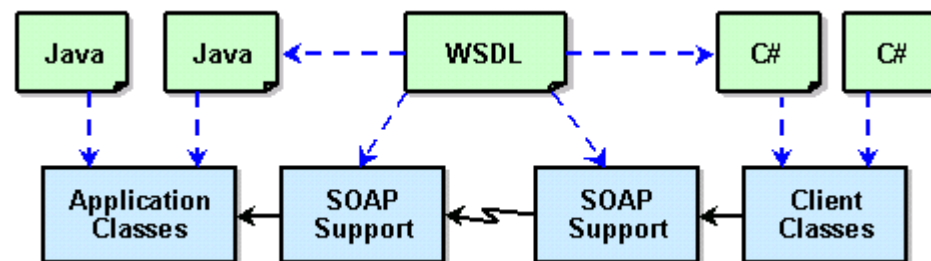
```

# WSDL first vs Language first

## Language first



## WSDL first



Source: <http://www.xml.com/pub/a/ws/2003/07/22/wsdlfirst.html>

# WSDL 2.0 structure

`<wsdl:description xmlns:wsdl="http://www.w3.org/ns/wsdl">`

`<wsdl:types/>`

XML schema and type definitions

`<wsdl:interface/>`

web service operations

`<wsdl:binding/>`

defines method of communication

`<wsdl:service/>`

associates a URI with a specific interface and binding

`</wsdl:description>`

# types

```
<wsdl:types>
  <xs:schema targetNamespace="http://rostlab.org/webservice/disulfinder">

    <xs:import namespace="http://rostlab.org/disulfinder/output"
schemaLocation="http://il2r-tbl.informatik.tu-
muenchen.de/~alex/disulfinder_output.xsd"/>

    <xs:element name="fastaSequence" type="xs:string" />
    <xs:element ref="output:disulfideBridgePrediction" />
    <xs:element name="getVersionResponse" type="xs:string" />
    <xs:element name="disulfinderError" type="xs:string" />

  </xs:schema>

</wsdl:types>
```

# interface

```
<wsdl:interface name="DisulfinderWebAPI">
  <wsdl:fault name="disulfinderFault" element="disulfinderError"/>
  <wsdl:operation name="getDsBonds" pattern="http://www.w3.org/ns/wsdl/in-out" wsdlx:safe="true">
    <wsdl:documentation>Get cystein binding states and ...</wsdl:documentation>
    <wsdl:input element="fastaSequence"/>
    <wsdl:output element="output:disulfiderBridgePrediction"/>
    <wsdl:outfault ref="disulfinderFault"/>
  </wsdl:operation>

  <wsdl:operation name="getVersion" pattern="http://www.w3.org/ns/wsdl/in-out" wsdlx:safe="true">
    <wsdl:documentation>Get method version.</wsdl:documentation>
    <wsdl:input element="#none"/>
    <wsdl:output element="getVersionResponse"/>
    <wsdl:outfault ref="disulfinderFault"/>
  </wsdl:operation>
</wsdl:interface>
```

# binding

```
<wsdl:binding name="DisulfinderHTTPBinding" interface="DisulfinderWebAPI"
type="http://www.w3.org/ns/wsdl/http" whttp:methodDefault="GET">

  <wsdl:documentation>The RESTful HTTP binding...</wsdl:documentation>
  <wsdl:fault ref="disulfinderFault" whttp:code="500"/><!--Internal Server Error-->

  <wsdl:operation ref="getDsBonds" whttp:location="getDsBonds" whttp:method="POST"/>
  <!-- http://alex.tbl/webservice/disulfinder/RESTdisulfinder.php/getDsBonds -->

  <wsdl:operation ref="getVersion" whttp:location="getVersion" whttp:method="GET"/>
  <!-- http://alex.tbl/webservice/disulfinder/RESTdisulfinder.php/getVersion -->
</wsdl:binding>
```

# service

```
<wsdl:service name="DisulfinderREST"  
interface="DisulfinderWebAPI">
```

```
  <wsdl:documentation>Disulfinder cystein disulfide...  
  </wsdl:documentation>
```

```
  <wsdl:endpoint  
    name="DisulfinderHTTPEndpoint"  
    binding="DisulfinderHTTPBinding"  
    address="http://alex.tbl:5000/" />
```

```
</wsdl:service>
```



# Differences for WSDL 1.1 and SOAP

<description>

<types/>

<interface/>

<binding/>

<service/>

</description>

<definitions>

<types/>

<message/>

<portType/>

<binding/>

<service/>

</definitions>

# messages

```
<wsdl:message name="getDisulfinderRequestMsg">  
  <wsdl:part name="parameters" element="fastaSequence"/>  
</wsdl:message>
```

```
<wsdl:message name="getDisulfinderResponseMsg">  
  <wsdl:part name="parameters" element="output:disulfideBridgePrediction"/>  
</wsdl:message>
```

```
<wsdl:message name="getVersionMsg">  
  <wsdl:part name="parameters" element="getVersion"/>  
</wsdl:message>
```

```
<wsdl:message name="getVersionResponseMsg">  
  <wsdl:part name="parameters" element="getVersionResponse"/>  
</wsdl:message>
```

# portType

```
<wsdl:portType name="DisulfinderWebAPI">
  <wsdl:documentation>This document describes the Disulfinder Web service... </wsdl:documentation>

  <wsdl:operation name="getDsBonds">
    <wsdl:documentation>Get cystein ...</wsdl:documentation>
    <wsdl:input message="getDisulfinderMsg"/>
    <wsdl:output message="getDisulfinderResponseMsg"/>
  </wsdl:operation>

  <wsdl:operation name="getVersion">
    <wsdl:documentation>Get method version.</wsdl:documentation>
    <wsdl:input message="getVersionMsg"/>
    <wsdl:output message="getVersionResponseMsg"/>
  </wsdl:operation>

</wsdl:portType>
```

# binding

```
<wsdl:binding name="DisulfinderSOAPBinding" type="DisulfinderWebAPI">

  <wsdl:documentation>The SOAP binding for the Disulfinder service.</wsdl:documentation>
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>

  <wsdl:operation name="getDsBonds">
    <soap:operation soapAction="http://alex.tbl:5000/getDsBonds"/>
    <wsdl:input><soap:body use="literal"/></wsdl:input>
    <wsdl:output><soap:body use="literal"/></wsdl:output>
  </wsdl:operation>

  <wsdl:operation name="getVersion">
    <soap:operation soapAction="http://alex.tbl:5000/getVersion"/>
    <wsdl:input><soap:body use="literal"/></wsdl:input>
    <wsdl:output><soap:body use="literal"/></wsdl:output>
  </wsdl:operation>

</wsdl:binding>
```

# REST vs SOAP

Consider "Martin Lawrence" as your data

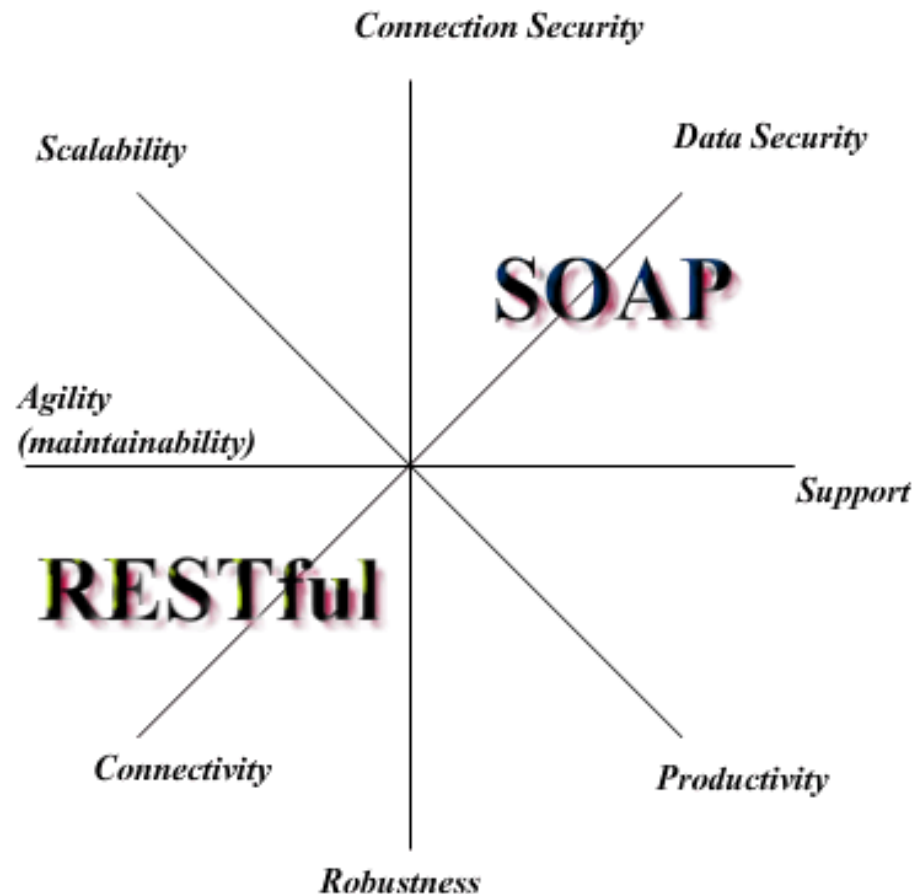
## SOAP



## REST



# REST vs SOAP



Source: <http://nicolas-zozol.developpez.com/tutorial/java/rest-jsp-english/>

# The implementation

- Use WSDL first-enabled framework
  - e.g. ZSI for python
    - Client  
`wsdl2py --complexType --file=ZonedLocation.wsdl`
    - Server  
`wsdl2dispatch --extended --file=ZonedLocation.wsdl`
- Look at WSDL and implement it yourself

# REST Server class

```
from flask import Flask,request
import time
import subprocess
import os

app = Flask(__name__)
#creates a folder with th current time in ms and returns its
name
def timeFolder(prefix):
    millis = str(int(round(time.time() * 1000)))
    folName = "/" .join([prefix.rstrip("/"),millis])
    if not os.path.exists(folName):
        os.makedirs(folName)
    return folName

@app.route('/getDsBonds',methods=['POST'])
#handle getDsBonds request
def getDsBonds():
    #create tmp folders
    folder = timeFolder("tmp")
    os.makedirs(folder+"/out")

    with open(folder+"/sequence.fasta","w+") as f:
        f.write(request.data)
```

```
#call disulfinder
shellCmd = " ".join(["disulfinder","-f",folder+"/sequence.fasta","-o",folder+"/out","-r",folder,"-d /data/uniprot_sprot","-c"])
subprocess.call(shellCmd,shell=True)
#retrieve output file
outFile = folder+"/out/" + list(os.listdir(folder+"/out"))[0]
shellCmd = " ".join(["./disulfxml","-i",outFile,"-o",folder+"/sequence.xml"])

subprocess.call(shellCmd,shell=True)
#open xml output File and return contents
with open(folder+"/sequence.xml") as f:
    out = f.read()
    return out

@app.route('/getVersion')
def getVersion():
    p = subprocess.Popen("disulfinder
--version",stdout=subprocess.PIPE,stderr=subprocess.PIPE,shell=True)
    out,err = p.communicate()
    return err

if __name__ == '__main__':
    app.debug=True
    app.run(host='0.0.0.0')
```



# REST Client class

```
import requests
from optparse import OptionParser
import sys

def main():
    parser = OptionParser()
    parser.add_option("-i", "--input",
,dest="inFile")
    parser.add_option("-o", "--output",
,dest="outFile")
    options,args = parser.parse_args()

    #define Web service URI
    serviceURI = "http://alex.tbl:5000"
    #read fasta from file or stdin
    if options.inFile != None:
        with open(options.inFile) as f:
            fasta = f.read()
    else:
        fasta = sys.stdin.read()

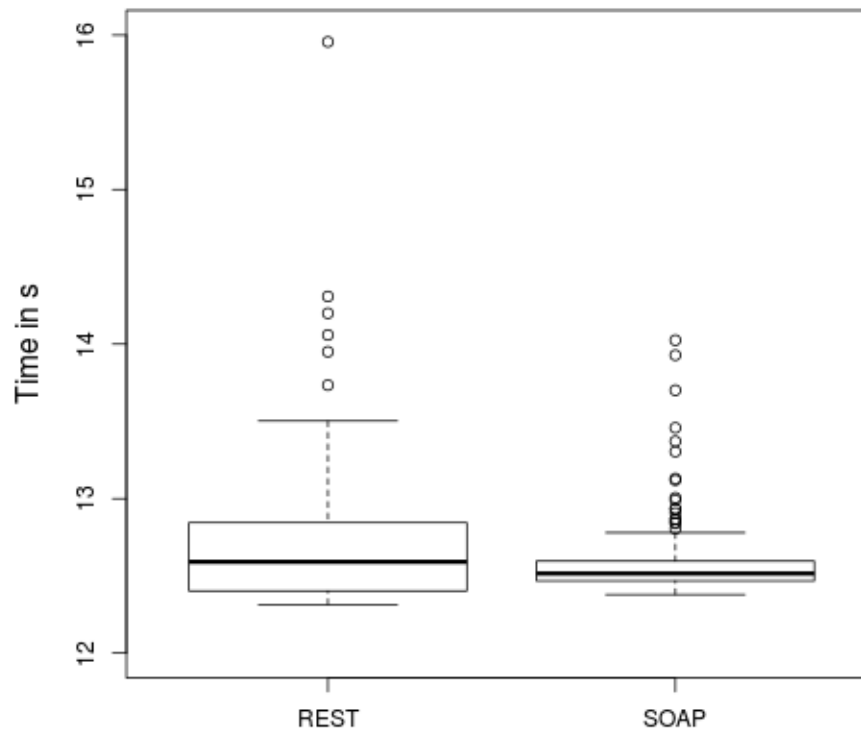
    #send fasta file per POST to Webservice
    res=
    requests.post(serviceURI+"/getDsBonds",fasta)
    #retrieve content
    msg = res.content

    #write to file
    with open(options.outFile,"w+") as f:
        f.write(msg)

if __name__ == "__main__":
    main()
```

# Testing

100 runs with the same protein from different VM

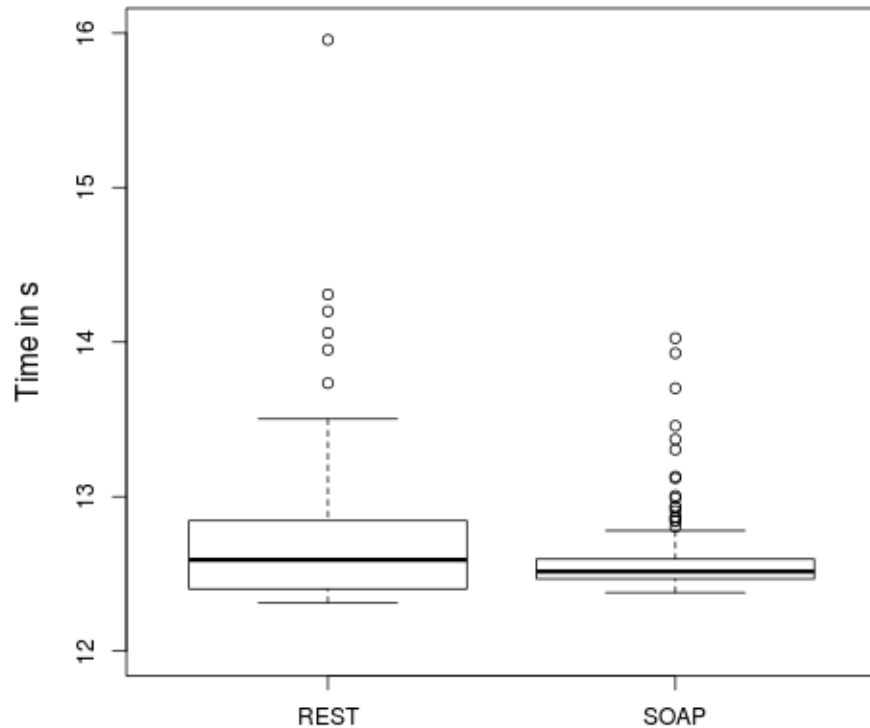


python client  
PHP Server

python client  
python server

# Testing

100 runs with the same protein from different VM



Missing:

- Local runtime
- Network traffic comparison

# Web services and security

- SOAP + WS security
  - Industry standard for web services
  - end-to-end encryption
- REST + HTTPS
  - Encryption while on the internet
  - Not encrypted between HTTP and application

# Web services and security

- SOAP + WS security
  - end-to-end encryption
  - Industry standard for web services
- REST + HTTPS
  - Encryption while on the internet
  - Not encrypted between HTTP and application

**BUT:** WS can be implemented for REST as well

And don't forget ...



**Web services**



**are lovely!**



# Resources

- WSDL-first explanation  
<http://www.xml.com/pub/a/ws/2003/07/22/wsdlfirst.html>
- Python ZSI cookbook  
<http://pywebsvcs.sourceforge.net/cookbook.pdf>
- WSDL specification from W3  
<http://www.w3.org/TR/wsdl>
- IBM whitepaper on WSDL2.0 and REST  
<http://www.ibm.com/developerworks/webservices/library/ws-restwsdl/>
- <http://rhizomik.net/html/~roberto/thesis/html/WebTechnologies.html>
- Python Flask Microframework  
<http://flask.pocoo.org/>
- Python request HTTP module  
<http://docs.python-requests.org/en/latest/>  
accessed on 08-07-2013



Thank you for your attention!  
If you have any questions, feel free to ask

# WSDL 2.0 structure

`<wsdl:description xmlns:wsdl="http://www.w3.org/ns/wsdl">`

`<wsdl:types/>`

XML schema element and type definitions

`<wsdl:interface/>`

web service operations, including the specific input and output

`<wsdl:binding/>`

defines how a client can communicate with the Web service

`<wsdl:service/>`

associates an address for the Web service with a specific interface and binding

`</wsdl:description>`